Attorney Docket No.: 2870-001-CIP

System and Method for Using Multiple Communication Protocols in Memory Limited Processors

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority under 35 U.S.C. § 119(e) from provisional application number 60/394,207 filed July 5, 2002 and application no. 10/354,527 filed January 30, 2003 both of which applications are incorporated by reference herein, in their entirety, for all purposes.

FIELD OF THE INVENTION

[0002] The present invention relates generally to implementation of telecommunication protocols. More particularly, the present invention relates to the implementation of large and/or multiple telecommunication protocols utilizing memory-limited processors.

BACKGROUND OF THE INVENTION

[0003] Less than thirty years ago, the term "telecommunications" connoted making and receiving telephone calls over the public switched telephone network (PSTN) built by AT&T. Today, telecommunications means transporting data representing voice, video, text, and instructions over wired and wireless digital networks such as the Internet.

[0004] Within the PSTN telephony environment the equipment needed to support the telecommunications infrastructure was centralized at a telephone company "central office" so that Customer Premises Equipment (CPE) could be limited to simple telephones. The nature of modern digital networks is to decentralize many functions and capabilities thus requiring more complex CPE to provide access. However, subscribers expect newer digital telecommunications to be usable with the same ease as the traditional telephone and at low cost. This expectation dictates that the digital network interfaces and associated protocols be compact and unobtrusive, implemented inexpensively, and require little in the way of subscriber interaction.

[0005] The complexity of implementing a CPE for digital telecommunications

comes primarily from the need to implement the protocols used to organize information sent over digital networks. These protocols evolved in rich computing environments with many computing resources (e.g. CPU power for computation; memory for data and program storage). Additionally, the protocols evolved quickly (i.e. in months or years, vs. the traditional PSTN years or decades), reflecting knowledge gained from actual application. The CPE for digital communications of today must, therefore, have an effective way to deal with protocol implementation within the restricted computational resources dictated by the CPE's restricted hardware cost.

[0006] What would be useful is a system and method for implementing multiple digital telecommunication protocols on a reduced hardware CPE that is not limited to any specific protocol.

SUMMARY OF THE INVENTION

[0007] An embodiment of the present invention is a telecommunications gateway that implements telecommunication protocols using a telecommunication protocol engine (TPE). Telecommunication protocols comprise multiple digital networking protocols (e.g., Session Initiation Protocol (SIP), H.323, DHCP, TCP/IP and STUN protocol) and telephony protocols. However, the present invention is not so limited. As will be apparent to those skilled in the art, any protocol that facilitates telecommunications over digital networks (both between digital devices, a digital device and an analog device, and between analog devices) may be implemented by the TPE without departing from the scope of the present invention.

[0008] In this embodiment, the TPE is implemented using inexpensive, memory limited microprocessors and inexpensive flash memory. However, this is not meant as a limitation. As will be apparent to those skilled in the art, the present invention may be implemented in other computing contexts without departing from the scope of the present invention.

[0009] Therefore, an aspect of the present invention is an implementation of a telecommunication protocol engine (TPE) using a memory limited microprocessor and flash memory.

[0010] Another aspect of the present invention is an implementation of a (TPE) using is a "virtual machine" that executes instructions from flash memory.

[0011] Another aspect of the present invention is the representation of telecommunication protocols as Finite State Machine (FSM) abstractions.

[0012] Still another aspect of the present invention is the implementation of a virtual machine to support FSMs used to express protocol implementation.

[0013] A further aspect of the present invention is the implementation of FSMs using virtual machine instructions stored in a flash memory, wherein the virtual machine instructions represent telecommunication protocols implemented as states, instructions, and transitions.

[0014] Yet another aspect of the present invention is a CPE Control Protocol that specifies how an end user can control the behavior of the CPE using a standard telephone that may be connected directly to the CPE or that accesses the CPE remotely.

[0015] An aspect of the present invention is a telephony gateway comprising a TPE.

[0016] Another aspect of the present invention is the implementation of the Session Initiation Protocol (SIP), H.323 protocol, DHCP and STUN protocol as FSM.

[0017] These and other aspects of the present invention will become apparent from a review of the general and detailed descriptions that follow. An embodiment of the present invention is a telecommunications gateway that implements multiple digital networking protocols using a telecommunication protocol engine (TPE). In this embodiment, the TPE is implemented using inexpensive, memory limited microprocessors and inexpensive flash memory. However, this is not meant as a limitation. As will be apparent to those skilled in the art, the present invention may be implemented in other computing contexts without departing from the scope of the present invention.

[0018] A finite state machine (FSM) execution facility is implemented using virtual

machine instructions located in the flash memory. The "state" of a given instance of a FSM is located in the RAM and accessed by the microprocessor. As the virtual machine executes instructions from the flash memory, it modifies the FSM state in RAM along with accessing other facilities of the microprocessor and other software resources.

[0019] Another embodiment of the present invention comprises a method for representing a protocol as a FSM using an extension of the C++ programming language. Specifically, an FSM specification may be created on a development computer using C++ with a specialized library. The result is a program that, when executed on the development computer, produces virtual machine instructions that can be loaded into the flash memory of the TPE. The virtual machine within the TPE microprocessor then uses these instructions as described above. While this embodiment uses the C++ programming language and a C++ library, the present invention is not so limited. As will be appreciated by those skilled in the art, other programming languages (and related libraries) may be used to produce virtual machine instructions that define an FSM.

[0020] An additional aspect of the present invention is the specification of a CPE Control Protocol that is implemented using the technique described above. This protocol specifies how an end user can control the behavior of the CPE using a standard telephone that may be connected directly to the CPE or accessing the CPE remotely by "calling" over either a VoIP or PSTN connection. This protocol allows the user to direct the CPE to place a local telephone to VoIP call; a local telephone to local PSTN call or a received VoIP call routed to the local PSTN based call. Additionally, this protocol allows the user to modify other operations of the CPE.

[0021] The CPE Control Protocol receives input from the user via the standard telephone touch-tone keypad. Specifically, the user enters a pound-sign (#), a sequence of digits or stars identifying the operation with any related data and a terminating pound-sign (#) indicating the end of user input. The CPE Control Protocol communicates with the user via one or more facilities depending on the originating location of the command. These include: flashing of the LEDs on the

CPE, generation of tones played over the telephone; voice commands played over the telephone; placing a call back to the telephone and using the "Caller ID" mechanism to present alpha numeric data via the telephones Caller ID display facility.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] **Figure 1A** illustrates a functional block diagram of a telecommunication protocol engine according to an embodiment of the present invention.

[0023] **Figure 1B** illustrates a functional block diagram of a telecommunication protocol according to an embodiment of the present invention.

[0024] **Figure 1C** illustrates a function block diagram showing data flows in a telecommunication protocol engine according to an embodiment of the present invention.

[0025] **Figure 2** illustrates a flowchart of a process for implementing a telecommunication according to an embodiment of the present invention.

[0026] **Figure 3** illustrates means for expressing a FSM according to an embodiment of the present invention.

[0027] **Figure 4** illustrates a VoIP telecommunication between a caller and a called party from the perspective of the caller according to an embodiment of the present invention.

[0028] **Figure 5** illustrates a script created using a scripting language to implement the protocol illustrated in **Figure 4**.

[0029] **Figure 6** illustrates a block diagram of a telephony gateway according to an embodiment of the present invention.

[0030] **Figure 7** illustrates a service offering according to an embodiment of the present invention

[0031] **Figure 8** illustrates a service provider gateway according to an embodiment of the present invention

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0032] An embodiment of the present invention is a telecommunications gateway that implements multiple protocols using a telecommunication protocol engine (TPE). In this embodiment, the TPE is implemented using an inexpensive, memory limited microprocessors and inexpensive flash memory. A finite state machine (FSM) execution facility is implemented in firmware using virtual machine instructions located in the flash memory. The "state" of a given instance of a FSM is located in the RAM and accessed by the microprocessor. As the virtual machine executes instructions from the flash memory, it modifies the FSM state in RAM along with accessing other facilities of the microprocessor and other software resources. However, this is not meant as a limitation. As will be apparent to those skilled in the art, the present invention may be implemented in other computing contexts without departing from the scope of the present invention.

[0033] Referring to **Figure 1A**, a block diagram of a TPE **100** according to an embodiment of the present invention is illustrated. The TPE comprises microprocessor **102** and flash memory **140**. Microprocessor **102** comprises CPU **120**, RAM **115,** and firmware **105** in which a virtual machine **125** resides. The CPU **120** runs the virtual machine **125** and communicates with the flash memory **140** over data bus **110.** In the preferred embodiment, data bus **110** is a parallel bus, but this is not meant as a limitation. In another embodiment, data bus **110** is a serial bus. The flash memory holds protocol templates for protocol A **145,** protocol B **155,** and protocol N **160** and data relating to each protocol. Each template comprises virtual machine instructions defining one or more FSMs implementing a particular protocol, illustrated for protocol A only **150.**

[0034] The template provides the specifications for the one or more FSMs that represent a single protocol. Each FSM specification is used to generate a set of virtual machine instructions (illustrated in **Figure 1A** for protocol A **150**) that define a FSM implementing all or a portion of a particular protocol.

[0035] **Figure 1B** illustrates a functional block diagram of a telecommunication protocol according to an embodiment of the present invention. Protocol A **145** comprises virtual machine instructions **150**. The virtual machine instructions **150**

for protocol A **145** define finite state machine 1 **160**, finite state machine 2 **161**, and finite state machine N **162**. Thus, a single protocol may comprise more than one finite state machine. Telecommunication protocols comprise multiple digital networking protocols (e.g., Session Initiation Protocol (SIP), H.323, DHCP, TCP/IP and STUN protocol) and telephony protocols. However, the present invention is not so limited. As will be apparent to those skilled in the art, any protocol that facilitates telecommunications over digital networks (both between digital devices, a digital device and an analog device, and between analog devices) may be implemented by the TPE without departing from the scope of the present invention.

[0036] The virtual machine instructions are read and executed on demand by a virtual machine that resides in firmware **105**. Since the microprocessor **120** (see **Figures 1A** and **1C**) retrieves instructions in response to a request from the virtual machine **125**, only a tiny portion of an entire protocol specification (specifically one virtual machine instruction and its associated operands) is stored in RAM **115** at any given time. One advantage of this arrangement is that the RAM **115** contains only the currently executing virtual machine instruction and data, and a few miscellaneous data structures needed for representing the current FSM states (also fixed in size). The content of the flash memory **140** – which can be quite sizeable if it implements a number of complex protocols – does not utilizes any significant portion of RAM **115**. The size of the virtual machine instructions **150** is limited by the amount of flash memory **140** available, and has no significant impact on the RAM **115** resources. This allows complex protocols to be implemented on very low cost microcontroller architectures in which program and data memory is very limited but flash memory is plentiful.

[0037] **Figure 1C** illustrates a function block diagram showing data flows in a telecommunication protocol engine according to an embodiment of the present invention. The bus references have been deleted for clarity. A protocol creation system **175** produces virtual machine instructions **150**. In an embodiment of the present invention, protocol creation system **175** comprises a C++ development environment that is used to encode an FSM specification for a protocol as human readable text. The resulting C++ program is then executed to generate the virtual

machine code that is store in the flash memory **140** via I/O ports **165** and I/O bus **170**. While this embodiment uses the C++ programming language and a C++ library, the present invention is not so limited. As will be appreciated by those skilled in the art, other programming languages (and related libraries) may be used to produce virtual machine instructions that define an FSM.

[0038] A virtual machine instruction and its operands **150** are retrieved by the CPU **120** at the direction of the virtual machine **125** and stored in RAM **115** for execution. Virtual machine **125** then executes the instruction that was retrieved.

[0039] A telecommunication protocol is implemented in accordance with the process embodiment illustrated in **Figure 2**. A call is made to the TPE to implement a telecommunication protocol **200**. The call is processed **205** by the microprocessor via the TPE software, which determines which telecommunication protocol is to be implemented **210** and where the protocol template resides in flash memory **215.** FSM data is then created in RAM and initialized to represent the state of the FSM used to implement the protocol **220**. The virtual machine **225** then starts executing instructions from a location within the flash memory based upon the FSM data.

[0040] The classic definition of a FSM is a collection of states. When a FSM is being executed, it is said to be "in" a specific state waiting for an event. When an event occurs, the FSM definition asserts a next state to be entered. When a state is entered a set of actions may performed, then the FSM waits in that state for the next event. This FSM operational model is implemented directly by the virtual machine described above in reference to **Figure 1B** and supported by the specialized virtual machine instructions generated from the FSM that expresses the protocol. Specifically, the State entry actions are expressed as specialized virtual machine instructions. Additionally, the virtual machine is capable of executing multiple FSM at the same time be they multiple instances of one FSM definition or different FSM definitions. It is through this facility that the TPE supports multiple protocols concurrently.

[0041] The virtual machine in this embodiment is specially designed to operate with events modeled as "tokens" so that it can respond to both physical events

identified by the microprocessor firmware or logical events generated by other FSM being concurrently executed in a uniform manner. The representation of "tokens" and their management further enhances the TPE to operate on very low cost microprocessor architectures in which program and data memory is very limited.

[0042] Virtual machine instructions are generated using a "translator" that receives human-readable syntax and translates this syntax to FSM instructions. To facilitate the specification of a protocol in human readable form and support its maintenance as the protocol evolves over time, the translator and virtual machine support the concept of shared state entry instructions through function or macros. These are collections of virtual machine instructions that reside in the flash memory, along with FSM specifications. As required, the virtual machine can execute these special collections of instructions upon demand.

[0043] **Figure 3** illustrates a means for expressing a FSM according to an embodiment of the present invention. Referring to **Figure 3**, a function name is associated with a purpose. These function name/purpose pairs are used to author protocol implementations via FSM definitions. A function name/purpose pair (referred to as an "Advocate") corresponds to a facility "known" to the virtual machine that is implemented in the firmware. While this exemplary embodiment uses the C++ programming language, the present invention is not so limited. As will be appreciated by those skilled in the art, other programming languages may be used to produce virtual machine instructions that define an FSM.

[0044] The C++ definition of each Advocate comprises code that produces the appropriate virtual machine instructions that will be placed into the flash memory. Note that some Advocates take parameters. The task of setting up virtual machine instructions for accessing and updating these parameters is also performed by C++ code within the body of the Advocate.

[0045] **Figure 4** illustrates a VoIP telecommunication protocol according to an embodiment of the present invention between a caller and a called party from the perspective of the caller. **Figure 5** illustrates a script according to an embodiment of the present invention created using the scripting language **(Figure 3)** to

implement the protocol illustrated in **Figure 4.** The caller initiates a call **400** causing the FSM to send a message to the called party **410.** Referring to **Figure 5**, the initial state of the FSM is "CALL INITIATED." The FSM then receives signaling messages via the network from the called party indicating the response (status) **420** of the called party. If the called party is busy, the FSM receives a CALLED PARTY BUSY condition (referred to as an event) **430,** executes a PLAY BUSY TONE command (an action) **460**, and enters the state "BUSY TONE" **475**. Similarly, if there is no response from the called party and the call times out **440**, the FSM receives a TIME OUT condition **440**, executes a PLAY FAST BUSY TONE command **470**, and enters the state "FAST BUSY TONE" **485**. If the call is delivered, the FSM receives a CALL DELIVERED condition **430**, executes both an "Init Vocoder" and "Send (CONNECT_ACK)" instruction **465**, and enters the state "Voice" **480**.

[0046] **Figure 6** illustrates a block diagram of a telephony gateway **600** according to an embodiment of the present invention. A digital signal processor (DSP) **603** communicates with a CPU **610** over a host interface/API **605**. Power to the telephony gateway **600** is provided via A power supply **615**. DSP **600** communicates with a subscriber line interface (SLI) **620** that offers connection to consumer premises equipment through an external jack 1 **625**. The SLI **620** interfaces with customer premises equipment, such as a telephone. DSP **600** communicates with a data access arrangement (DAA) **635** that offers connection to the public switched telephone network (PSTN) through an external jack 2 **637.**

[0047] The CPU **610** is connected to a RAM **650**, to a flash memory **640**, and to a firmware unit **660**. Together, the CPU **610,** the RAM 650, the flash memory **640,** and the firmware unit comprise a telecommunication protocol engine as described in the context of **Figure 1**. The CPU **610** also provides an interface to I/O ports **630**. In an embodiment of the present invention, I/O ports **630** comprises a serial interface and an Ethernet interface. The I/O ports are accessible through an external jack 3 **645.** It is anticipated that as RAM becomes cheaper and more capable of enhanced storage that firmware unit **660** can also reside within RAM **650.**

[0048] **Figure 7** illustrates a service offering according to an embodiment of the present invention. Referring to **Figure 7**, a service provider network **700** comprises a service provider proxy **703**, a service provider gateway **705** and connectivity to the public switched telephone network (PSTN) **740**, ISP-A network **710**, and ISP-B network **715**. A telephony gateway-A **720** is in communication with ISP-A network **710**. Telephone-A **725** is connected to telephony gateway-A **720**. Similarly, a telephony gateway-B **730** is in communication with ISP-B network **715**. Telephone-A **735** is connected to telephony gateway-B **730**.

[0049] The connection between a telephony gateway (**720**,**730**) and its associated ISP network (**710**, **715**) is made by means known in the art. By way of illustration and not as a limitation, telephony gateway-A **720** is in communication with ISP-A network **710** using DSL connection. Telephony gateway-B **730** is in communication with ISP-B network **715** via a dialup connection. It is noteworthy that no general-purpose computer is required to establish communication between a telephony gateway (**720**, **730**) and the service provider gateway **705**.

[0050] Both telephony gateway-A **720** and telephony gateway-B **730** are registered with service provider network **700**. Telephony gateway-A **720** may place a call to telephony gateway-B **730** by interacting with the service provider network **700**. The service provider proxy **703** then contacts telephony gateway-B **730** on behalf of telephony gateway-A **720** to facilitate call setup (also known in the art as signaling). Once the signaling has been completed, telephony gateway-A **720** interacts directly with telephony gateway-B **730** passing information until the call is terminated. When the call is terminated, call tear down singling is performed through the service provider network proxy **703**.

[0051] Both telephony gateway-A **720** and telephony gateway-B **730** are registered with service provider network **700**. Service provider network **700** is also connected to the PSTN **740**, which is connected to telephone-C **745** and telephone-D **750**. A call placed by telephony gateway-A **720** to telephone-C via the PSTN **740** would involve call setup and tear down signaling between telephony gateway-A **720**, server provider proxy **703**, and service provider gateway **705**.

[0052] A call placed over a service provider network **700** is routed by service

provider gateway **705**. In an embodiment of the present invention, the protocol used by a telephony gateway (**720**, **730**) and the routing is controlled by a number dialed to initiate a telephone call via a CPE Control protocol. By way of illustration and not as a limitation, a call placed from telephone-A **725** to telephone-B **735** is an "on-network" call, meaning both the calling party and receiving party are using registered telephony gateways (**720**, **730**). In this embodiment of the present invention, the telephone numbers associated with registered telephony gateways begin with the same prefix, for example 777. In this embodiment, the calling party using telephone-A **725** presses #777(plus the remaining telephone number digits)# on telephone-A **725** (note that the starting and ending pound-sings (#) reflect the requirements of the CPE Control protocol). Service provider gateway **705** determines from the prefix preceding the remaining telephone number digits that the call is on-network and connects telephone-A **725** to telephone-B **735** over a service provider network **700**.

[0053] By contrast, if telephone-A **725** places a call to telephone-C **745**, the call is dialed using the standard prefix (1+areacode+number) again bracketed by the pound-sings required by the CPE Control protocol (e.g. #16503288459#).

[0054] As will be apparent to those skilled in the art, other signaling conventions may be used to route telephone calls without departing from the scope of the present invention.

[0055] In an embodiment of the present invention, a service provider gateway is a self-contained system for providing telephone communications using telephony gateways as embodied herein. **Figure 8** illustrates a service provider gateway according to an embodiment of the present invention.

[0056] Referring to **Figure 8**, service provider gateway **800** comprises network systems **805** and protocol handling systems **815**. These systems permit the service provider gateway to communicate with various networks using protocols required by the networks or required by the type of communication service being provided. Interactive voice response (IVR) software **830**, in combination with billing system **835** and data management and storage systems **850** gather billing data and automate billing questions. Authentication of users is managed by

authentication system **825.**

[0057] In yet another embodiment of the present invention, a service provider gateway **800** according an embodiment of the present invention as described in reference to **Figure 8** is licensed for use by a third party service provider (TPSP). In one embodiment, the TPSP is permitted under the license to provide service to subscribers using a telephony gateway as described previously wherein the service is private labeled in the name of the TPSP. In an alternate embodiment, the TPSP acquires franchise rights to provide services to subscribers in the name of the licensor. In still another embodiment, the TPSP is permitted to offer access to a service provider network operated by the licensor in the name of the TPSP.

[0058] In yet another embodiment of the present invention, the telephony gateway is configured to receive communications from a remote location via a telephone call (either from the PSTN or a wireless service provider). The communications may be used to configure the gateway or to initiate a call from the gateway to a remote communication device. In this latter embodiment, the gateway additionally functions as a bridge between the incoming calling device and the remote communication device. The gateway answers the incoming calling, the CPE Control protocol accepts user input (e.g. a authentication PIN, star and target phone number: #65432*7771234567#) necessary to call the remote receiving device, and then places a VoIP call to the remote communication device.

[0059] In another embodiment of the present invention, two telephony gateways are connected to first and second communication devices respectively that are in communication with each other. The first communication device sends a "hook-flash" signal to the first telephony gateway. The first telephony gateway suspends the communication with the second telephony gateway and enables caller access to the CPE Control protocol. Using this protocol the user directs the CPE to initiate a three-way call to another phone. The CPE Control protocol will notify the CPE Control Protocol on the second device that a three-way call has been initiated. The three-way connection comprises sharing data from one party with the other two parties on the call. In this manner, a three-way connection is established and maintained without external mixing devices or the need to deploy a media gateway

in the VoIP system.

[0060] A telephony gateway, CPE Control protocol and a telecommunication protocol engine and method have now been illustrated. As described herein, the telecommunication protocol engine and method results in significant reduction in on-chip memory requirements and permits the use of otherwise memory limited microprocessors. It will also be understood that the invention may be embodied in other specific forms without departing from the scope of the invention disclosed and that the examples and embodiments described herein are in all respects illustrative and not restrictive. Those skilled in the art of the present invention will recognize that other embodiments using the concepts described herein are also possible.